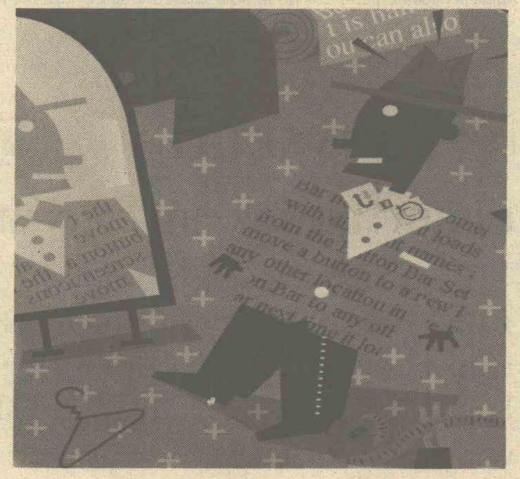
มุมนักเล่น

SPREAD SHEET

ศรันย์ ไมตรีเวช

5 เหตุผลในการเลือก EXCEL 5.0 พัฒนาแอพพลิเคชัน

มุมมองหนึ่งในการพิจารณาเลือกใช้โปรแกรมเอ็กเซล 5.0 ในการสร้างแอพพลิเคชันของคุณเอง



ขอเท้าความถึงบทความตอนที่แล้วในวารสารไมโครฯ ฉบับที่ 112 ซึ่งอยู่ในบทความเรื่อง *เอ็กเซล 5.0 กับการสร้างแอพพลิเคชันของคุณเอง* สักนิดนะครับ เราทราบกันว่าข้อเด่นข้อดีของเอ็กเซล 5.0 ที่ทำให้คุณน่าจะเลือกใช้มันเป็นเครื่องมือพัฒนาแอพพลิเคชันของตัวเอง ว่ามี 5 ประการคือ

- มีออปเจ็กต์มากมายที่ควบคุมด้วยโค้ดได้
- ภาษาโปรแกรมอันลือลั่นอย่างวิชวลเบสิก
- สนับสนุน OLE 2.0
- สนับสนุน WOSA โปรโตคอลถ่ายข้อมูล (ODBC, MAPI, และ TAPI)
- เป็นสเปรดชีตที่รันได้บนทุกแพลตฟอร์ม

ในตอนนี้เราจะมาดูรายละเอียดของแต่ละข้อ โดยจะให้แต่ละข้อเป็นหัวเรื่องใหญ่โดยเฉพาะ

1.เอ็กเซลมีออปเจ็กต์มากมายที่ควบคุมด้วยโค้ดได้

เอ็กเซลถูกซอยกลุ่มก้อนใหญ่ ๆ ของมันออกเป็นชิ้นเล็กชิ้นน้อย พูดอีกนัยหนึ่งก็คือ ระบบทั้งเครือถูกแบ่งเป็นองค์ประกอบย่อยต่าง ๆ มากมาย แต่ละองค์ประกอบนี้สามารถควบคุมได้ด้วยโค้ดของเอ็กเซลเอง ซึ่งทางไมโครซอฟต์แถลงว่างานนี้ลงทุนลงแรงกันไม่อั้น จ้างทีมงานผู้เชี่ยวชาญเขียนโค้ดกันหลายร้อยคนในเวลาหลายปีที่ผ่านมา ว่ากันว่าใช้อัลกอริธึมที่เยี่ยมที่สุดในโลกด้วย ผลที่ตามมาคือความสะดวกสบายสำหรับเราท่านในการพัฒนาโปรแกรมไว้ใช้งานเองด้วยงบจำกัดอย่างคุ้มค่า เรื่องมันดีอยู่สองอย่างครับ คือ

1.1 โปรแกรมเมอร์สามารถรวมออปเจ็กต์เข้าไปไว้ในโค้ด

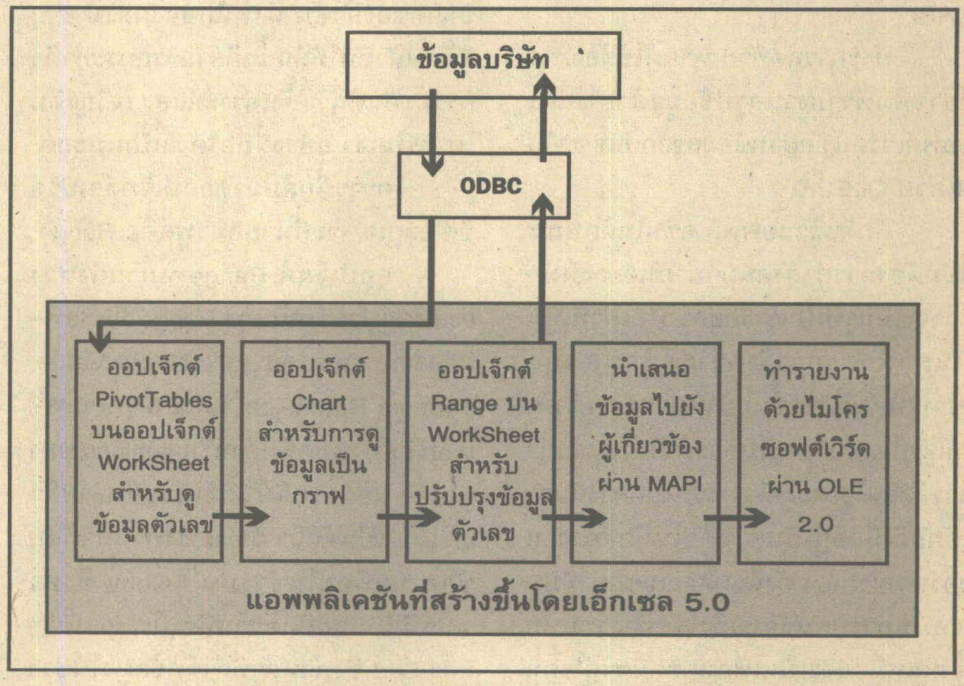
ในอดีตที่ผ่านมา หากโปรแกรมเมอร์ต้องการสร้างอะไร เช่น แผนภูมิ (chart)

ในแอพพลิเคชันล่ะก็ คงได้แต่กัมหน้ากัมตาเขียนโมดูลเกี่ยวกับกราฟขึ้นมาสักก้อนหนึ่ง เพราะงานประเภทนี้จะต้องใช้เวลามาก กว่าความถูกต้องเรื่องสูตรเรื่องตัวเลขจะลงตัวกันเต็มร้อย แต่ด้วยเอ็กเซล 5.0 โปรแกรมเมอร์ไม่ต้องเขียนโค้ดเลย แค่ใช้ออปเจ็กต์แผนภูมิของเอ็กเซล (MS Excel Chart Object) เพียงผ่านข้อมูลไปให้ออปเจ็กต์ดังกล่าวพร้อมคำสั่งง่ายๆเล็กน้อย เอ็กเซลก็จะรับภาระจัดการให้ทั้งหมด ทั้งเรื่องการคำนวณเกี่ยวกับตัวเลข การวาดกราฟ และจิปาถะทั้งหลายแหล่ ทำให้โปรแกรมเมอร์ไม่ต้องเปลืองแรง เปลืองเวลา แถมยังได้แอพพลิเคชันที่ดูดีและน่าใช้ดีกว่าเดิมอีกด้วย (สำหรับคุณที่เคยคุ้นกับวิชวลเบสิก คงนึกเทียบได้กับคอนโทรลกราฟนั่นแหละครับ หน้าที่เรามีแค่บอกว่าต้องการกราฟประเภทไหน ข้อมูลมีอะไรบ้าง นอกนั้นวิชวลเบสิกดำเนินการให้หมด เราไม่ต้องคำนวณอะไรเลยจริงๆ)

เอ็กเซลประกอบด้วยออปเจ็กต์ที่มีความสามารถอื่นๆ มากมาย ซึ่งแต่ละอย่างใช้งานง่ายทั้งนั้น เป็นต้นว่าออปเจ็กต์ชื่อ PivotTable ซึ่งทางไมโครซอฟต์คิดว่าน่าจะเป็นออปเจ็กต์มหัศจรรย์ที่สุด เพราะมีกลไกที่สามารถนำข้อมูลปริมาณมหาศาลจากดาต้าเบสมาใส่ไว้ในสเปรดชีตได้อย่างสมบูรณ์แบบ เทคโนโลยีของ PivotTable ทำให้การเรียกข้อมูลอย่างรวดเร็วเป็นไปได้ และให้การติดต่อกับผู้ใช้ดำเนินไปด้วยความสะดวกง่ายดาย

ออปเจ็กต์อื่นที่ควรกล่าวถึงก็มี เวอร์กบุ๊ค เวอร์กชีต เมนู ทูลบาร์ ไดอะล็อกชีต ลิสต์บอกซ์ ออปชันบ็อกซ์ และเช็คบ็อกซ์ แต่ละอย่างเหมาะกับการนำไปใช้ในการพัฒนาแอพพลิเคชันระดับมืออาชีพทั้งสิ้น การให้ออปเจ็กต์มาอย่างครอบคลุมในการนำไปใช้งานด้านต่างๆนี้ ช่วยให้เราสามารถใช้เอ็กเซลไปประยุกต์กับงานต่างๆได้อย่างกว้างขวาง บางบริษัทถึงกับเอาไปทำ EIS (Enterprise Information System) ซึ่งเป็นระบบที่ทำหน้าที่ให้ความหมายกับข้อมูลบริษัทจำนวนมาก และอันเนื่องมาจากเอ็กเซลสนับสนุน ODBC มันจึงสามารถอ่านข้อมูลจากดาต้าเบสต่างๆได้หลายฟอร์แมต และเมื่อข้อมูลถูกนำเข้ามาอยู่ในเอ็กเซลเรียบร้อย ก็จะมีเครื่องไม้เครื่องมือที่มีประสิทธิภาพหลายต่อหลายชิ้นที่จะนำมาใช้วิเคราะห์และดำเนินการกับข้อมูลต่างๆ เหล่านั้น เป็นต้นว่า PivotTables ฟังก์ชันเกี่ยวกับเวอร์กชีตมากกว่า 400 ฟังก์ชัน รูทีนเกี่ยวกับการลำดับข้อมูล อัลกอริธึมในการรวมผลย่อย (subtotal) การทำแผนภูมิ และอีกมากมายซึ่งไม่ได้กล่าวถึงก่อนอื่นมาลองดูไดอะแกรมในรูปที่ 1 ซึ่ง EIS นำเอ็กเซลไปประยุกต์ใช้กับงานกันว่ามันอยู่ในรูปลักษณะใด

ด้วยการประยุกต์ใช้อย่างที่เห็นในไดอะแกรมนี้ ข้อมูลจะถูกดึงจากดาต้าเบสผ่าน ODBC และใช้ในการนำPivot Tables มาตั้งบนเวอร์กชีตด้วยพร้อมกัน โดยที่ PivotTables นี้ยังให้ผู้ใช้เห็นหลายส่วนของข้อมูลในแง่ที่แตกต่างกันไปได้อีกด้วย และหลังจากที่ได้ทัศนาข้อมูลแบบตัวเลขกันเรียบร้อยแล้ว ก็มาดูข้อมูลในรูปแบบใหม่ที่เข้าใจง่ายกว่าเดิมเป็นรูปกราฟผ่านออปเจ็กต์ Chart กัน แน่นอนครับมันหมายความว่าคุณสามารถเชื่อมสองออปเจ็กต์คือ PivotTables เข้ากับ Chart ได้ เพื่อที่จะได้ดูข้อมูลในแง่มุมที่แตกต่างกันไปเช่นเดียวกับการดูข้อมูลแบบตัวเลข Chart จะรับข้อมูลจาก PivotTables โดยตรง การทำอย่างนี้ได้นั้นก็ด้วยการใช้คอนโทรล on-sheet ซึ่งประกอบด้วยลิสต์บ็อกซ์ ออปชันบ็อกซ์และเช็คบ็อกซ์ คอนโทรลพวกนี้จะอยู่บน Chart และ



รูปที่ 1 แสดงไดอะแกรมการนำ EXCEL 5.0 มาใช้สร้างแอพพลิเคชัน

สามารถลิงค์ไปยัง PivotTables ซึ่งที่จริงก็คือการควบคุม PivotTables นั่นเอง

หลังจากได้ดูข้อมูลทั้งแบบตัวเลขและแบบกราฟแล้ว ผู้ใช้อาจต้องการเปลี่ยนแปลงข้อมูล อันนี้ก็เจอบ่อยในงานเกี่ยวกับงบประมาณอะไรทำนองนั้น สิ่งที่ผู้ใช้ต้องทำก็คือเข้าไปในเวอร์กชีตซึ่งทำการรวบรวมสรุปข้อมูลต่างๆที่แสดงอยู่ใน PivotTable ด้วยหน้าจอเวอร์กชีตนี้ ผู้ใช้สามารถเปลี่ยนแปลงข้อมูลที่อยู่ในดาต้า-เบสซึ่งกำลังติดต่อกันกับเอ็กเซลอยู่ได้โดยผ่านออปเจ็กต์ชื่อว่า Range ดังที่เห็นในไดอะแกรมรูปที่ 1

วิชวลเบสิกสำหรับแอพพลิเคชันได้ให้คำสั่งไว้เยอะมากในอันที่จะดำเนินการโยกย้ายข้อมูลระหว่างเอ็กเซลและดาต้าเบสที่มี ODBC สนับสนุน หลังจากทำการเปลี่ยนแปลงข้อมูลบนเวอร์กชีตเรียบร้อยแล้ว ผู้ใช้สามารถย้ายไปยังส่วนอื่นของแอพพลิเคชัน ซึ่งก็คือกล่องที่สี่ในไดอะแกรม แล้วส่งสรุปข้อมูลไปยังผู้เกี่ยวข้องต่างๆในบริษัทด้วยระบบอีเมล์ผ่าน MAPI

ประการสุดท้าย หากผู้ใช้ต้องการก็อาจพิมพ์รายงานสรุปข้อมูลด้วยไมโครซอฟต์เวิร์ด โดยผ่านข้อมูลจากเอ็กเซลไปให้ด้วย OLE2.0

ปกติแล้วแอพพลิเคชันในลักษณะนี้มักต้องการกำลังคนมาทำกันอย่างหนักกว่าจะเป็นรูปเป็นร่างออกมาได้เท่าที่เห็น ในส่วนของการวิเคราะห์ข้อมูล เอ็กเซลทำให้มันง่ายขึ้นกว่าที่ผ่านมา บริษัทใหญ่ๆในอเมริกาเริ่มตระหนักกันแล้วว่าการตัดสินใจโดยผู้จัดการระดับกลางนั้นจะได้ผลดีเมื่ออยู่เป็นส่วนตัวในห้องทำงานของพวกเขาเอง ดังนั้นเมื่อเอ็กเซลมามีบทบาทในการป้อนข้อมูลถึงโต๊ะทำงานแบบนี้ ก็ดูเหมือนจะเป็นเพื่อนคู่คิดมิตรคู่ใจคนใหม่ของบริษัทไปได้อย่างลอยลำ

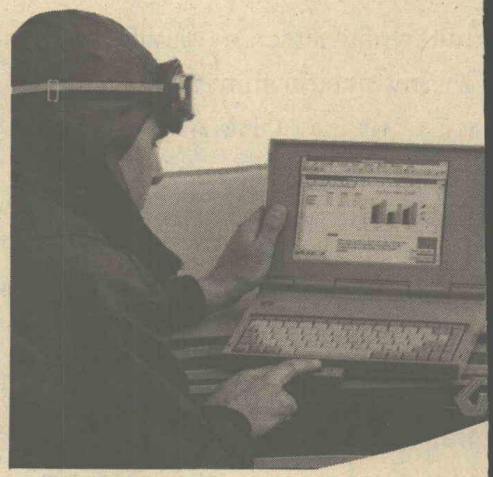
เอาล่ะครับ ทีนี้ลองมาดูส่วนของเทคโนโลยีอันเป็นเบื้องหลังการรวมตัวเป็นหนึ่งเดียวกันระหว่างออปเจ็กต์กับโค้ด

1.2 ออปเจ็กต์ของเอ็กเซลมีพรอพเพอร์ตี้ส์และเมธอดหลายตัว

ขอเรียนให้ทราบสำหรับท่านที่ยังไม่รู้ พรอพเพอร์ตี้ส์ (Properties) คือแง่มุมต่างๆของความเป็นออปเจ็กต์ตัวหนึ่งอย่างเช่น สี ขนาด ชื่อเสียงเรียงนามอะไรทำนองนั้น เราอาจกำหนดค่าให้พรอพเพอร์ตี้หนึ่งๆได้ ส่วนเมธอด (Methods) นั้นคือฟังก์ชันที่ทำให้อยู่ในรูป extension ของออปเจ็กต์ พอใส่ extension นี้ลงข้างหลังชื่อออปเจ็กต์ก็จะเกิดการกระทำอย่างหนึ่งขึ้นทันที จนดูเหมือนกับว่ามันคือการกระทำของตัวออปเจ็กต์เอง ยกตัวอย่างง่ายๆ เอาต้นไม้ต้นหนึ่งเป็นออปเจ็กต์ ต้นไม้นั้นมีพรอพเพอร์ตี้ส์ เช่น ขนาดของลำต้น น้ำหนักของผลไม้ อายุ สีสัน แล้วต้นไม้ต้นนั้นก็สามารถกระทำกิจกรรมเป็นต้นว่า สังเคราะห์แสง เจริญเติบโต ให้ร่มเงา อย่างนี้ถือได้ว่าเป็นเมธอด

คราวนี้กลับมาดูออปเจ็กต์สเปรดชีต โดยเฉพาะที่น่ายกมาพูดคือ Range

ออปเจ็กต์ Range หมายถึงช่วงเซลในสเปรดชีตนั่นเอง Range มีพรอพเพอร์ตี้ส์หลายตัว เช่น Name Value Formula RowHeight และอื่นๆ พรอพเพอร์ตี้ส์ทั้งหมดนี้มีค่าหรือ settings ของตนเอง นอกจากนี้มันก็มีเมธอดหลายตัวเช่นกัน เป็นต้นว่า Clear ซึ่งทำหน้าที่ลบข้อความที่อยู่ในช่วงนั้น ๆ Copy ซึ่งทำหน้าที่คัดลอกข้อความที่อยู่ในช่วงนั้นๆ และ Sort ซึ่งทำหน้าที่ลำดับข้อมูลในช่วงนั้นๆ



กระบวนการโค้ดออปเจ็กต์ของเอ็กเซลมักเกี่ยวข้องกับการกำหนดค่าหรือ settings ให้กับพรอพเพอร์ตี้ส์ หรือไม่ก็เรียกใช้เมธอดของพวกมัน ซึ่งการจะเขียนโค้ดดังกล่าวได้ก็ต้องผ่านภาษาแมคโครใหม่เอี่ยมของเอ็กเซล ซึ่งก็คือวิชวลเบสิกสำหรับแอพพลิเคชัน ยังไงล่ะครับ

2. เอ็กเซลมีภาษาแมคโครใหม่เอี่ยม : วิชวลเบสิกสำหรับแอพพลิเคชัน

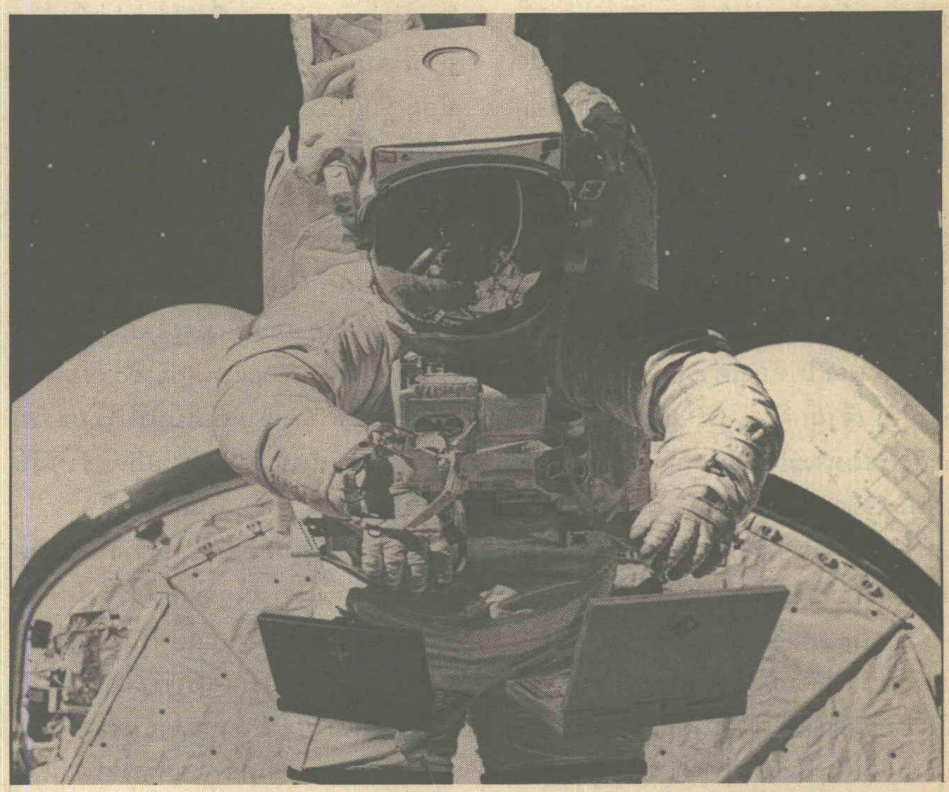
เอ็กเซล 5.0 เป็นผลิตภัณฑ์แรกที่ใช้ภาษาแมคโครใหม่เอี่ยมของไมโครซอฟต์ ซึ่งมีชื่อเป็นทางการว่าวิชวลเบสิกสำหรับแอพพลิเคชัน (Visual Basic for Applications) ภาษาใหม่นี้ที่จริงก็เอาอะไรต่ออะไรมาจากระบบของวิชวลเบสิกซึ่งกำลังฮือฮากันในขณะนี้ ซินแท็กซ์และเนื้อหาที่เกี่ยวกับโค้ดไม่ต่างกันเลย คุณๆที่เล่นวิชวลเบสิกกันถนัดมืออยู่แล้วก็จะพอใจยิ่งขึ้นเพราะคุณจะเสีย

เวลาเรียนรู้อะไรที่แตกต่างไปจากเดิมไม่เท่าไหร่

ในวงการนั้นยอมรับกันแล้วครับว่าวิชวลเบสิกเป็นเครื่องมือสร้างแอพพลิเคชันบนวินโดวส์ที่มหัศจรรย์ที่สุด ใช้ง่าย ทำได้เร็ว และมีความสามารถอย่างเหลือเชื่อ ไม่ว่าจะทำโปรโตไทป์หรือของจริง และบัดนี้ผู้พัฒนาแอพพลิเคชันบนเอ็กเซล 5.0 ก็สามารถใช้เครื่องมือมหัศจรรย์ดังกล่าวมาร่วมกับออปเจ็กต์ทั้งหลายในหัวข้อก่อนหน้านี้ เพื่อสร้างแอพพลิเคชันอันทันสมัย รวดเร็ว และดูแลได้ง่าย

เท้าความไปถึงภาษาแมคโครกันหน่อย ใครเคยพัฒนาแอพพลิเคชันด้วยคำสั่งแมคโครคงประจักษ์ถึงข้อด้อยในความเป็น Keystroke-based หรือ Function-based ของมัน กล่าวคือภาษาที่เป็น Keystroke-based นั้นจะขาดความสามารถหลากหลายและความยืดหยุ่นต่างๆนานา มันจะเกี่ยวข้องกับการเลียนแบบการเคาะคีย์ซึ่งผู้ใช้สเปรดชีตจะใช้ในการทำงานเท่านั้น ส่วนภาษาที่เป็น Function-based ซึ่งแม้จะมีความสามารถในการทำงานสูงกว่า Keystroke-based แต่ก็มีด้านกลับคือนำมาใช้ยาก แต่ละฟังก์ชันมักมีอาร์กิวเมนต์ที่ค่อนข้างจะยาว เมื่อมีค่าอะไรให้ต้องจัดการมากๆ ซึ่งอันนี้เป็นเรื่องยากที่โปรแกรมเมอร์อย่างเราจะไปจำกันไม่หวาดไม่ไหว ว่าซินแท็กซ์ของแต่ละคำสั่งเป็นอย่างไร แถมยังต้องป้อนข้อมูลอ้างอิงให้มันใช้ได้ถนัดๆ อีกต่างหาก

ในการแนะนำตัววิชวลเบสิกสำหรับแอพพลิเคชันของเอ็กเซลครั้งนี้ ทางไมโครซอฟต์เสนอภาษาแมคโครเชิงวัตถุ (Object-Oriented Macro Language) เป็นครั้งแรก กล่าวคือจะให้ข้อได้เปรียบหลายแง่มุมที่เหนือกว่า Keystroke-based และ Function-based ดังนี้



- ประการแรกสุดก็คือ ความสามารถในงานเหนือกว่าภาษาแมคโครแบบอื่นใด ด้วยการมีพรอพเพอร์ตี้ส์และเมธอด ซึ่งเป็นของคู่กันกับการโปรแกรมเชิงวัตถุ ซึ่งใช้ได้ในสภาพแวดล้อมของเอ็กเซล

- ประการที่สองคือ ความง่ายที่จะใช้ เพราะสถาปัตยกรรมเชิงวัตถุนั้นให้ผลเป็นความง่ายต่อการอ้างหรือกำหนดค่าให้พรอพเพอร์ตี้ส์กับเมธอดด้วยภาษาอังกฤษอันเรียบง่ายตรงไปตรงมา ไม่ยุ่งยากซับซ้อนต่อการทำความเข้าใจแต่อย่างใด

- ประการสุดท้ายคือมีสภาพแวดล้อมที่เอื้ออำนวยการแก้ไขและการดีบักเป็นอย่างสูง คือให้เครื่องไม้เครื่องมือในการนี้มากมายอย่างชนิดที่หาไม่ได้แม้ในภาษาระดับสูงก็ตาม

จากข้อได้เปรียบเหล่านี้ ก็เรียกได้ว่าไมโครซอฟต์เอ็กเซลนำเสนอสิ่งใหม่ในอันที่จะช่วยผู้พัฒนาให้ทำงานสำเร็จได้ง่ายขึ้นและเร็วขึ้นจริง ไม่ต้องแคลงใจรอถามผู้เชี่ยวชาญที่ไหนอีกถ้ามองกันเฉพาะแง่นี้ลองมาดูอย่างละเอียดลงไปก็ได้ครับหากต้องการข้อมูลเชิงเปรียบเทียบว่าข้อได้เปรียบระหว่างภาษาแมคโครแบบใหม่มันเหนือชั้นกว่าของเก่าอย่างไร

2.1 ในแง่ของความสามารถขั้นสูง

กล่าวถึงแล้วว่าโดยสาระสำคัญๆ ความเด่นของลักษณะโปรแกรมมุ่งไปที่การใช้พรอพเพอร์ตี้ส์กับเมธอด และนอกจากนั้นยังมีการเชื่อมคำสั่งพื้นฐานแบบที่นิยมใช้กันในการเขียนโค้ดทั่วไปแยกอีกต่างหาก แถมด้วยการเชื่อมคำสั่งแบบใหม่ที่หาไม่ได้จากภาษาแมคโครในอดีตที่ผ่านมา ยกตัวอย่างเช่น ประโยคคำสั่ง For-Each อันนี้จะให้แบบวิธีในการกระทำชนิดเดียวกันบนคอลเลคชันของออปเจ็กต์แบบเดียวกันทั้งหมด

วิชวลเบสิกสำหรับแอพพลิเคชัน

ยังให้คุณใช้ตัวแปรได้ทุกชนิด รวมทั้ง user-defined แต่ละชนิดจะกำหนดให้เป็นตัวแปรคงที่ อะเรย์ หรือฟังก์ชันแมคโครอย่างไรก็ได้

ในส่วนของภาษาแมคโครชนิด Keystroke-based นั้น อย่างดีก็ให้กลไกในการทำงานเป็นชุดโดยอัตโนมัติแทนผู้ใช้ซึ่งจะต้องเลือกเมนูคำสั่งเอาเอง พูดง่ายๆว่าผู้ใช้ไม่ต้องเคาะเอาเองก็เท่านั้น ดังนั้นมันจึงถูกจำกัดความสามารถเท่าที่คำสั่งในเมนูจะมีให้ แต่สถาปัตยกรรมเชิงวัตถุในวิชวลเบสิกสำหรับแอพพลิเคชันมันให้อะไรยิ่งไปกว่านั้นมาก คือโปรแกรมเมอร์จะสามารถดำเนินการและควบคุมองค์ประกอบต่างๆของสเปรดชีตโดยตรงไม่ต้องเข้าไปอิงอะไรกับโครงสร้างเมนูแม้แต่น้อยผลลัพธ์ที่เกิดขึ้นไม่แต่จะเป็นข้อได้เปรียบแต่ยังเพิ่มระดับความยืดหยุ่นให้กับการออกแบบแอพพลิเคชันอีกด้วย แถมอีกอย่างครับ วิชวลเบสิกให้คุณเข้าถึงองค์-ประกอบที่สำคัญๆหลายอย่างของแอพ-พลิเคชัน ซึ่งบางทีไม่เป็นที่ปรากฏอยู่ในเมนูให้แก่ผู้ใช้ด้วยซ้ำ และในส่วนของภาษาแมคโครชนิด Function-based แม้จะดีเหมือนภาษาเชิงวัตถุในแง่ความสามารถที่สูงกว่า Keystroke-based แต่ก็ยังมีข้อจำกัดอยู่ในตัวรูปแบบของมัน กล่าวคือการติดตั้งภาษาแมคโครชนิดนี้จะทำได้ต่อเมื่อองค์ประกอบของสภาพแวดล้อมในการพัฒนาเข้ากันได้กับข้อจำกัดของฟังก์ชัน เรื่องของเรื่องคือภาษาแบบ Funcition-based จะอิงกลุ่ม Verbs-Functions ซึ่งผลกระทบในทางปฏิบัติก็คือมันเป็นการกระทำซึ่งถูกดำเนินโดยองค์ประกอบของสเปรดชีต อันจะยังให้เกิดข้อจำกัดทางความสามารถของภาษา ลองนึกดูก็แล้วกันครับว่าถ้าคุณต้องพูดภาษาอะไรที่มีคำกิริยาและคำนามน้อยๆ มันจะให้ความรู้สึกติดขัดในการใช้ภาษาขนาดไหน คุณจะใช้ภาษาสื่อสารอย่างไรด้วยการผสมคำนามกับกิริยาที่มีอยู่จำนวนน้อย รูปแบบประโยคที่สร้างถูกจำกัดทำให้รู้สึกอึดอัด

ฉันใดก็ฉันนั้น ภาษาเชิง Function-based ซึ่งมีจำนวนฟังก์ชันอยู่น้อยทำให้เราประยุกต์ใช้กับเวอร์กชีตไม่ได้หลากหลายอะไรนัก และเพื่อที่จะจัดการกับเรื่องนี้ ภาษาเชิง Function-basedจึงมีแนวโน้มในการสร้างฟังก์ชันเป็นไปโดยธรรมชาติ ฟังก์ชันหนึ่งควรประยุกต์ใช้ได้กับองค์ประกอบมากมายในสเปรดชีต

คงเห็นนะครับว่าอย่างนี้มันลดประสิทธิภาพในการโค้ดและการรันโค้ดไปด้วย ทีนี้ลองหันมาดูภาษาเชิงวัตถุอย่างวิชวลเบสิกสำหรับแอพพลิเคชัน ซึ่งมีลักษณะเน้นไปในส่วนประกอบต่างๆของสเปรดชีต(ตามสไตล์พรอพเพอร์ตี้ส์และเมธอด) ไม่ได้เน้นในเรื่องของการกระทำซึ่งเกิดขึ้นกับหรือเกิดโดยองค์ประกอบเหล่านั้น ผลก็คือ เราสามารถมีออปเจ็กต์ได้ไม่มีขีดจำกัด แต่ละตัวก็มีพรอพเพอร์ตี้ส์และเมธอดเป็นของตนเอง ซึ่งอย่างนี้ก็กระทบกับประสิทธิภาพในการเขียนโค้ดและการรันโค้ดไปด้วยอย่างแน่นอน ออปเจ็กต์ของสเปรดชีตพร้อมด้วยพรอพเพอร์ตี้ส์และเมธอดของใครของมันจะทำให้เกิดการรวมกันเป็นหนึ่งเดียวในระบบที่มีประสิทธิภาพยิ่งทั้งในการสร้างและการใช้

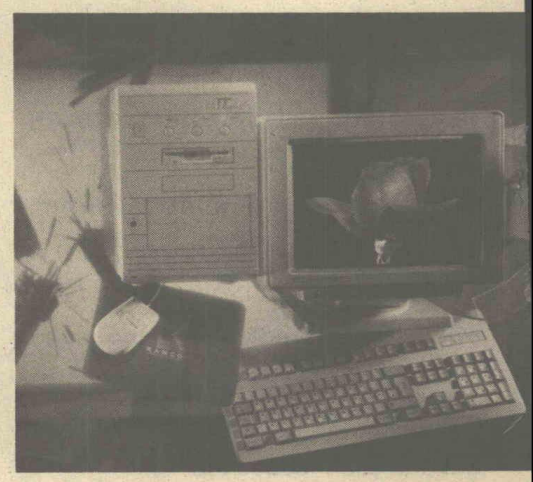
2.2 ในแง่ของความง่ายในการใช้

ด้วยวิชวลเบสิกสำหรับแอพพลิเคชันนี้ ออปเจ็กต์ของเอ็กเซลจะถูกจัดเรียงเป็นลำดับชั้น (Hierarchical Arrangement) กล่าวคือออปเจ็กต์พวกหนึ่งๆจะถูกบรรจุอยู่ด้วยออปเจ็กต์อื่นๆ (หากคุณใช้วิชวลเบสิกอยู่แล้วคงเข้าใจคอนเสิปต์นี้ง่ายขึ้น) ตัวอย่างง่ายที่สุดขอให้นึกถึงการอ้างคอนโทรลข้ามฟอร์ม คุณต้องระบุชื่อฟอร์ม แล้วใส่จุดตามด้วยชื่อคอนโทรล นี่ก็เป็นรูปแบบหนึ่งของการมีออปเจ็กต์อยู่ในออปเจ็กต์ เพราะทั้งฟอร์มและคอนโทรลต่างก็เป็นออปเจ็กต์ด้วยกันทั้งสิ้น

การเจาะเข้าไปถึงออปเจ็กต์ที่ต้องการก็ใช้รูปแบบที่ง่ายดาย นั่นคือใช้เครื่องหมายจุด (.) เป็นตัวไต่ลำดับลงไปเหมือนกับที่คุณอาจเคยใช้มาบ้างแล้วในภาษาอื่น นอกจากเข้าถึงออปเจ็กต์ได้ คุณยังเข้าถึงพรอพเพอร์ตี้ส์และเมธอดของออปเจ็กต์ได้ด้วยวิธีใส่เครื่องหมายจุดแบบเดียวกันนี้อีกด้วย อย่างเช่นต้องการใส่ชื่อให้ออปเจ็กต์ Range ซึ่งพรอพเพอร์ตี้คือ Name โปรแกรมเมอร์ก็เพียงเขียนว่า

```
Application.Workbooks("Book1").
Worksheets("Sheet1").Range("A1").Name = "
CellA1"
```

ดังตัวอย่างนี้ เป็นการแสดงให้เห็นการเจาะลงไปถึงออปเจ็กต์ที่ต้องการโดยผ่านลำดับชั้นออปเจ็กต์ แล้วอ้างอิงพรอพเพอร์ตี้ของออปเจ็กต์สุดท้ายที่ต้องการมาดูกันทีละส่วนนะครับ ออปเจ็กต์ Applica-



tion เป็นออปเจ็กต์ที่เป็นสาธารณะที่สุด (เป็นโกลบอลที่สุด) ในเอ็กเซลคือมีออปเจ็กต์ทุกตัวในแอพพลิเคชันบรรจุอยู่ในตัวมัน

ตัวต่อมาคือ Workbooks("Book1") อันนี้ใช้อ้างถึงออปเจ็กต์เวอร์กบุ๊คอย่างในกรณีนี้มีวงเล็บ ("Book1") ก็หมายถึงการอ้างเวอร์กบุ๊คชื่อ Book1 นั่นเองสำหรับคุณที่เกิดความสงสัยเพราะยังไม่คุ้นกับเอ็กเซล ก็ขอขยายความอีกนิดว่า เวอร์กบุ๊คนี้เป็นตัวแทนของไฟล์ซึ่งถูกสร้างและอ่านได้ด้วยเอ็กเซล ดังนั้นชื่อ Book1 ตามตัวอย่างจึงเป็นชื่อไฟล์ซึ่งจะเห็นตรงกันในดิสค์เมื่อ Dir ดู

ตัวต่อมาคือ Worksheets("Sheet1") อันนี้ใช้อ้างถึงออปเจ็กต์เวอร์กชีต ซึ่งอยู่ในเวอร์กชีต ซึ่งในที่นี้มีชื่อว่า Sheet1 คงไม่สงสัยนะครับเวอร์กชีตก็คือตารางที่คุณเห็นนั่นเอง

ตัวสุดท้ายคือ Range("A1") อันนี้ใช้อ้างถึงออปเจ็กต์ Range ซึ่งอยู่ในเวอร์กชีต ชื่อบอกตรงตัวอยู่แล้วว่าหมายถึงเซลช่วงไหน หรือเซลเดี่ยวช่องไหน ส่วนที่เห็น Name ซึ่งผมเขียนไว้เป็นตัวพิมพ์หนาพิเศษนั้นไม่ใช่ออปเจ็กต์แล้ว แต่เป็นพรอพเพอร์ตี้ของออปเจ็กต์ ซึ่งก็มีความหมายว่า Range("A1") นี่คุณจะตั้งชื่อให้มันว่า CellA1

ทราบไว้นิดครับว่าผลที่ออกมาจากบรรทัดคำสั่งตามตัวอย่างนี้ สามารถทำได้เหมือนกันอีกแบบหนึ่งอย่างย่อ เช่น

```
Workbooks("Book1").worksheets("Sheet1").Range("A1").Name = "CellA1"
```

ข้อแตกต่างก็ไม่มีอะไรมาก แค่คำว่า Application มันหายไปเท่านั้น เรื่องของเรื่องก็คือหากคุณจะรันโค้ดด้วยเอ็กเซลเป็นการภายใน ก็ไม่จำเป็นต้องอ้างอิง Application ให้เสียเวลา ต่อเมื่อรันจากภายนอก อย่างนี้คำว่า Application จึงจะขาดไม่ได้ อันที่จริงถ้าย่อให้สั้นที่สุดเลยก็จะเหลือเพียง

```
Range("A1").Name = "CellA1"
```

คือตัด Application,Workbooks และ Worksheet ทิ้งให้หมด อันนี้ใช้ได้ต่อเมื่อเป็นกรณีที่คำสั่งถูกเอ็กซีคิวต์ภายในเอ็กเซล ขณะที่ Sheet1 (ออปเจ็กต์ Worksheets) ใน Book1 (ออป-เจ็กต์ Workbooks) กำลังทำงาน บรรทัดคำสั่งนี้ย่อลงต่อไม่ได้อีกแล้วนะครับ อย่าหาทางพยายามเพราะคำสั่งเกี่ยวกับออปเจ็กต์นั้นต้องการอย่างน้อยที่สุดคือชื่อออปเจ็กต์กับพรอพเพอร์ตี้ส์และค่าที่เราต้องการกำหนด

ตัวอย่างลักษณะนี้แหละครับชี้ให้เห็นถึงความเป็นการเขียนโค้ดเชิงวัตถุด้วยการใช้วิชวลเบสิกสำหรับแอพพลิเคชัน ลองดูตัวอย่างเปรียบเทียบระหว่างภาษาแมคโครเชิงวัตถุ Keystroke-based และ Function-based กันดูโดยการกำหนดชื่อให้ช่วงเซล A1

แบบเชิงวัตถุ

```
Range("A1").Name = "CellA1"
```

แบบ Keystroke-based

```
/mccella1~
```

แบบ Function-based

```
=DEFINE.NAME("cella1", "=Sheet1!R1C1")
```

จากการยกตัวอย่างมาให้ดูเพื่อเปรียบเทียบอย่างนี้ จะเห็นได้ว่าสิ่งที่อ่านแล้วเกิดความเข้าใจเป็นธรรมชาติมากที่สุดน่าจะได้แก่ตัวอย่างแรก ซึ่งก็คือภาษาเชิงวัตถุนั่นเอง การใช้เครื่องหมายจุดมาเป็นซินแท็กซ์ถือว่าคือพัฒนาก้าวสำคัญของการเขียนโค้ดเพื่อความเป็นกันเองระหว่างเครื่องกับคน

การป้อนค่าด้วยลักษณะ Keystroke-based นั้นก็คือการใช้คำสั่งแมคโครแบบเดียวกับ Lotus 1-2-3 บนดอส และบนวินโดวส์ หากคุณไม่คุ้นกับมัน บรรทัดคำสั่งที่เห็นค่อนข้างจะสับสนนั้น ก็แค่แทนการกดคีย์ของผู้ใช้ดีๆนี่แหละครับ คือเขาเรียกเมนูคำสั่งกันด้วยการกดคีย์ดังที่เห็น จะอธิบายเป็นตัวๆเลยก็ได้

- เครื่องหมาย / : เป็นอักขระที่จะใช้เข้าถึงเมนูหลัก คือกดแล้วจะเห็นเมนูหลักปรากฏขึ้นมา
- ตัวอักษร r : เป็นเมนูคำสั่งเลือกช่วงเซล (range)
- ตัวอักษร c : เป็นเมนูย่อยเลือกสร้าง (create) ช่วงเซล
- cella1 : เป็นชื่อที่เราใส่เข้าไปถือเป็น entry นั่นเอง
- เครื่องหมาย ~ : เป็นอักขระที่เราใช้แทนการเคาะปุ่ม Enter หมายถึงทำการใส่ entry เรียบร้อยแล้วนั่นเอง

อันที่จริงดูแล้วคำสั่งเชิง Keystroke-based นี้มันสั้นกะทัดรัดดีใช่ไหมครับ สั้นกว่าของวิชวลเบสิกสำหรับแอพพลิเคชันตั้งเยอะ แต่ถ้าจะอ่านเขียนแต่ละ

ทียุ่งยากซะมัด ถึงจะคุ้นมาแล้วสักแค่ไหนก็ยังต้องนั่งแกะกันตาเหล่อยู่ดี เรื่องมันอยู่ที่ตรงนี้คือ วิชวลเบสิกสำหรับแอพพลิเคชันต้องนั่งเขียนกันยาวกว่า แต่ทั้งอ่านทั้งเขียนมันง่ายต่อการรับรู้เพราะเป็นภาษาอังกฤษพื้นฐานที่ใครอ่านก็เข้าใจได้ แทนที่จะต้องมาจดจำเมนูคำสั่งกันให้ทะลุปรุโปร่งกว่าจะใช้งานคล่อง เราแทนเสียด้วยภาษาอังกฤษพื้นๆอย่างนี้แหละแค่จำชื่ออ็อบเจ็กต์ ชื่อพรอพเพอร์ตี้ส์ และชื่อเมธอด รวมทั้งค่ากำหนดที่เป็นไปได้ให้ขึ้นใจไม่กี่ตัวก็ลอยลำแล้ว

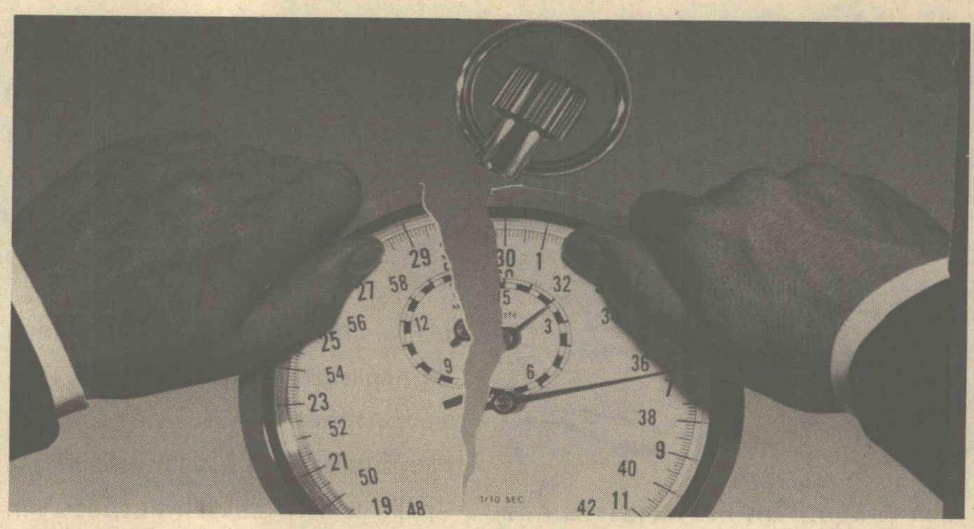
มาพูดถึงการป้อนค่าด้วยลักษณะ Function-based กัน อันนี้เป็นภาษาแมคโครของเอ็กเซลที่เก่าแค่เวอร์ชัน 4.0 ที่ผ่านมานี่เอง เรียกกันว่าภาษาแมคโครแบบ XLM โดยเราใช้ฟังก์ชัน DEFINE.NAME ซึ่งต้องการอาร์กิวเมนต์สองตัวตามหลัง นั่นคือชื่อหนึ่งและที่อยู่หนึ่งของ range นั่นๆ

ถึงยังไงวิธีนี้ก็ชวนพิศวงงงตาอย่างเดียวกับ Keystroke-based อยู่ดีจริงไหมครับ

สังเกตด้วยว่าโปรแกรมเมอร์ต้องเสียเวลาทำความรู้จักทั้งตัวฟังก์ชันเองและต้องแม่นเรื่องการปรากฏกับลำดับที่ถูกต้องของอาร์กิวเมนต์ รวมทั้งซินแท็กซ์ที่พลาดไม่ได้อีกด้วย

หากคุณอ่านตัวอย่างข้างต้นแล้วยอมรับว่ามันดูสับสนจริงๆดังว่า ก็ขอให้ทราบเถอะครับว่านี่เป็นการคัดเอาของง่ายมาโชว์แล้วจริงๆเวลาใช้จะยุ่งยากกว่านี้เยอะ นับตั้งแต่จำนวนอาร์กิวเมนต์มากมายเป็นหางว่าว ไปจนถึงลำดับและซินแท็กซ์ที่ต้องถูกต้องจริงๆ

ย้อนกลับไปดูวิชวลเบสิกสำหรับแอพพลิเคชัน พออ่านรูปคำสั่งแล้วมันช่างง่ายและดูดีเป็นธรรมชาติเสียจริงๆเห็นไหมครับว่าจะอ่านจะเขียนมัน "อยู่ในใจ"



และ "ออกมาจากหัว" ทั้งสิ้น แทบไม่มีความพลิกแพลงไปจากภาษามนุษย์ที่ใช้สื่อสารกันสักเท่าไหร่เลย นี่แหละครับเทคโนโลยีสูงขึ้นก็ต้องถูกมนุษย์ยุคใหม่ใช้ได้ง่ายขึ้นอย่างนี้ ไม่ใช่เทคโนโลยียิ่งสูงก็ยิ่งดูน่ากลัว ไม่กล้าเข้าไปแตะต้อง

2.3 ในแง่ของการแก้และตรวจหาจุดผิดพลาดของโค้ด

ความจริงไม่น่าเรียกวิชวลเบสิกสำหรับแอพพลิเคชันว่าภาษาแมคโครด้วยซ้ำนะครับ ทั้งนี้ก็เพราะคุณจะมีความรู้สึกเหมือนอยู่ในสภาพแวดล้อมของเครื่องมือพัฒนาโปรแกรมที่สมบูรณ์แบบได้มาตรฐาน อย่างที่จะกล่าวถึงนี้ก็คือมันมีเครื่องช่วยในการอีดิตและดีบักอยู่พร้อมมูล

มันอยู่ตรงจุดนี้ครับ วิชวลเบสิกสำหรับแอพพลิเคชันเป็นอีกโมดูลหนึ่งแยกต่างหากจากงานเปรดชีตบนเอ็กเซลเหมือนๆกับโค้ดในวิชวลเบสิก 3.0 หรืออื่นๆนั่นแหละ ไม่ได้แตกต่างอะไรกันเลย จากคอนเส็ปต์นี้จะเห็นได้ว่าเป็นคนละเรื่องกับภาษาแมคโครสมัยเก่าที่จะต้องถูกเก็บรวมกับชีตในโปรแกรมสเปรดชีต

Worksheet ซึ่งเป็นชีตที่เก็บและดำเนินการข้อมูลชนิดตัวเลข ธรรมชาติของมันไม่เหมาะกับการเก็บโปรแกรมหรอก เพราะมีความไม่สะดวกทางเทคนิคในอันที่จะปรับปรุงแก้ไขอะไร ดังนั้นไมโครซอฟต์จึงเล็งเห็นความจำเป็นที่จะต้องแยกส่วนโค้ดออกมาต่างหาก การปฏิวัติงานโค้ดในเอ็กเซลจึงจะเกิดขึ้นได้ กล่าวคือ วิชวลเบสิกสำหรับแอพพลิเคชันจะสนับสนุนทั้งการแก้ไขและการหาจุดผิดพลาดของโปรแกรมเช่นเดียวกับเครื่องมือพัฒนามาตรฐานทั่วไป ไม่ใช่แค่ภาษาแมคโครเก่าๆที่ถูกมองว่าช่วยอะไรไม่ได้เลยอีกต่อไป

ในวิชวลเบสิกสำหรับแอพพลิเคชัน จะมีคุณลักษณะช่วยแก้ไขและหาจุดผิดพลาดที่หาไม่ได้แน่นอนในโปรแกรมสเปรดชีตอื่นดังต่อไปนี้

Syntax Checking

มันเป็นเรื่องดีสำหรับเราใช่ไหมครับ ที่เมื่อเขียนโค้ดปุ๊บมันก็ตรวจซินแท็กซ์ปั๊บ รู้กันเดี๋ยวนั้นเลยว่ามีความผิดพลาดอะไรบ้างไหม โดยเฉพาะระยะแรกหากยังไม่เคยคุ้นกับการเขียนโค้ดด้วยวิชวลเบสิก ไม่ว่าจะมือเก่าหรือมือใหม่ ข้อผิดพลาดที่เกิดจากความไม่แม่นซินแท็กซ์จะมีความเป็นไปได้สูง การฟ้องว่าผิดหรือ

ถูกจะมีขึ้นหลังจากคุณเคาะ Enter หรือกดปุ่มลูกศรเปลี่ยนบรรทัดแต่ละครั้ง

นอกจากนี้ยังมีการทำอักษรตัวแรกของคีย์เวิร์ดสำคัญให้เป็นตัวพิมพ์ใหญ่ด้วย พวกนี้จะเป็นชื่อออปเจ็กต์ พรอพเพอร์ตี้ส์ และเมธอดที่วิชวลเบสิกรู้จักและแน่ใจอยู่แล้วว่าไม่มีทางถูกใช้แทนกัน เนื่องจากเป็นคำสงวน อย่างนี้นอกจากจะทำให้รูปโค้ดสวยงามเป็นระเบียบแล้ว ยังเป็น feedbackให้โปรแกรมเมอร์ทราบได้อีกนัยหนึ่งว่าการอ้างชื่อ พรอพเพอร์ตี้ส์หรือเมธอดมีความถูกต้องหรือไม่เพียงไร

เรื่องการฟ้องข้อผิดพลาดทางซินแท็กซ์นี่เป็นตัวเลือกที่โปรแกรมเมอร์ให้มีหรือไม่มีก็ได้ซึ่งจะอยู่ในรูปของเมสเสจบ็อกซ์ถ้ามีก็เป็นการรับประกันความผิดพลาดในแง่นี้ได้ดี แต่บางทีก็รำคาญเหมือนกันเวลาเรารู้อยู่แก่ใจว่าไม่ได้เป็นความผิดพลาดเผลอเรอ แต่เกิดพบข้อความที่อยากแก้ในบรรทัดอื่น แล้วใจร้อนเลื่อนเคอร์เซอร์ไปยังบรรทัดนั้นโดยไม่เขียนบรรทัดปัจจุบันให้เสร็จเรียบร้อยเสียก่อนอย่างนี้มันก็ฟ้องให้เสียเวลาแล้ว ก็เลือกเอาแล้วกันว่าอยากให้เป็นแบบไหน แต่ผมว่าความแน่นอนที่ได้มานี่มันคุ้มกับความรำคาญเล็กๆน้อยๆที่นานครั้งเกิดขึ้นเสียทีหนึ่งนะครับ

Color Coding

จะเป็นเรื่องดีไหมครับหากโค้ดของคุณถูกแบ่งพวกแบ่งคำด้วยสีให้เห็นชัดๆ ผลดีมันมีหลายอย่าง นั่นคือคุณแยกแยะได้ง่ายว่าอันไหนเป็นส่วนคำสั่ง อันไหนเป็นส่วนข้อมูล วิชวลเบสิกสำหรับแอพพลิเคชันมีฟีเจอร์นี้อยู่ครับ ก็ไม่แปลกประหลาดอะไร เดี๋ยวนี้ใครๆก็มีกันทั้งนั้น เอาเป็นว่าวิชวลเบสิกของเอ็กเซลไม่ตกมาตรฐานชาวบ้านเขาก็แล้วกัน

ไม่ใช่มีแค่สีสองสีให้เลือกนะครับ มีเป็นสิบๆสีทีเดียว และทุกพวกทุกกลุ่มของโค้ดสามารถแยกแยะได้ทั้งสิ้น ไม่ว่าจะเป็นคำสั่ง เนื้อความ คอมเมนต์ และอื่นๆ

Break Points

เป็นกลไกสำหรับเซ็ตจุดที่ต้องการให้หยุดในแมคโครของคุณเมื่อทำการเอกซีคิวต์โค้ดไปเรื่อยๆแล้วเจอเบรคพอยน์ปุ๊บ การเอ็กซีคิวต์ จะชะงักลงทันที และพร้อมกันนั้นก็จะแสดงโค้ดหน้าที่มีการเบรคโดยเฉพาะเน้นโฟกัสไปที่บรรทัดซึ่งเป็นจุดหยุด รวมทั้งหน้าต่างดีบักซึ่งช่วยในการหาข้อผิดพลาดโดยรายงานสถานะของตัวแปรและคำสั่งที่คุณคิดว่าน่ารู้ต่างๆ

Watch Variables

เป็นกลไกที่จะทำให้คุณกำหนดตัวแปรหรือคำสั่งตัวหนึ่งหรือชุดหนึ่ง เหมือนหมายตาไว้ว่าจะทราบค่าหรือสถานภาพของพวกมันเมื่อเกิดการหยุดเอ็กซีคิวต์ชั่วขณะ คุณจะเห็นการรายงานในหน้าต่างดีบักดังที่กล่าวถึงไปแล้วเมื่อครู่

การหมายตาตัวแปรหรือคำสั่งอันน่าจะเป็นต้นเหตุบักนี้ วิชวลเบสิกช่วยคุณไม่ได้หรอกครับ มันต้องเกิดจากประสบการณ์ของโปรแกรมเมอร์เอง หากตั้งข้อสันนิษฐานไว้ถูก คุณก็จะได้เห็นค่าหรือสถานภาพตัวแปร/คำสั่งมีความเพี้ยนไปจากความมุ่งหมาย ดังนั้นฟีเจอร์นี้เป็นประโยชน์สมบูรณ์ไม่ได้ด้วยตัวเอง แต่ต้องอาศัยการสันนิษฐานของโปรแกรมเมอร์ประกอบด้วย ซึ่งจะเป็นอย่างไรก็ลองผิดลองถูกได้หลายรูปแบบ คือเขาให้แม่บทมาอย่างนี้ เราไปประยุกต์อย่างไรก็เป็นเรื่องของทักษะเฉพาะตัวอีกที

Debug Window

อย่างที่กล่าวแล้วข้างบน หน้าต่างดีบักเป็นประโยชน์มากในการแยกเอา "กลุ่มบัก" ที่คุณหมายตาไว้ออกมาแสดง หน้าต่างนี้ถูกนำมาเมื่อไหร่ก็ได้ด้วยการกดปุ่ม F8 หรือคลิ๊กปุ่ม Step Macro บนทูลบาร์ของวิชวลเบสิก ภายในหน้าต่างดีบักนี้ จริงๆแล้วแบ่งออกเป็นสามส่วนด้วยกัน คือ

1. ส่วนโค้ด(Code Pane) คือหน้าต่างอีดิตโค้ดซึ่งคุณเขียนมันขึ้นมานั่นเอง

2. ส่วนแสดง(Watch Pane) คือส่วนที่คุณจะได้เห็นตัวแปรที่ต้องการรวมทั้งค่าในปัจจุบันของมัน

3. ส่วนรับคำสั่งปัจจุบัน(Immediate Pane) คือส่วนที่คุณสามารถออกคำสั่งใดๆ นอกตัวโค้ดเดิม คือเกิดคุณอยากทดลองให้มีผลลัพธ์อะไรสักอย่างด้วยคำสั่งสดๆเดี๋ยวนั้น ก็สามารถกระทำได้ที่นี่ เป็นต้นว่าอยากลองดูว่ามันจะเกิดอะไรขึ้นถ้าให้มีค่าตัวแปรต่างไปจากเก่า จะเห็นได้ว่านอกจากจะหาที่ผิดพลาดได้แล้ว ยังช่วยให้คุณเกิดความเข้าใจเหตุการณ์ที่ควรหรือไม่ควรเกิดได้อีกโสดหนึ่งด้วย

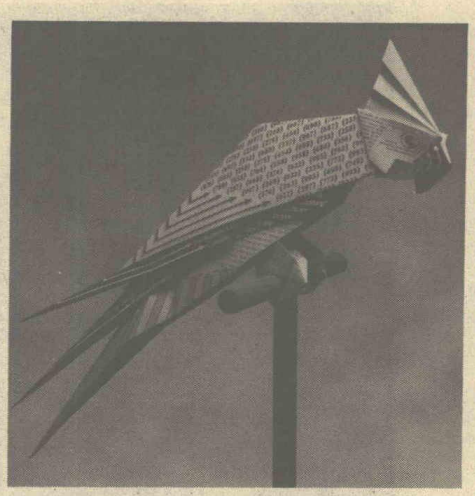
Object Browser

เป็นเครื่องมือที่มีความสามารถสูงในสภาพแวดล้อมของการปรับปรุงแก้ไขโค้ด โดยมันจะแสดงลิสต์และนำข้อมูลจากออปเจ็กต์ทุกตัวพร้อมทั้งพรอพเพอร์ตี้ส์และเมธอดที่เกี่ยวข้องมาให้กับเราได้

ออปเจ็กต์ที่ถูกแสดงโดย Object Browser นั้น นอกจากปรากฏได้ในเอ็กเซลแล้ว ยังไปให้เห็นได้ในแอพพลิเคชัน OLE 2.0บนระบบเดียวกันอีกด้วยยกตัวอย่างเช่นโปรแกรมเมอร์มีงานที่ต้องใช้ทั้งเอ็กเซลและไมโครซอฟต์โปรเจ็กต์ควบคู่กันไป ซึ่งต่างก็สนับสนุน OLE 2.0 และมีวิชวลเบสิกสำหรับแอพพลิเคชันด้วยกันทั้งสองฝ่าย Object Browser จะสามารถแสดงออปเจ็กต์ทั้งหมดของทั้งสองฝั่งให้คุณเห็นได้

On-line Help

รายละเอียดต่างๆเกี่ยวกับออปเจ็กต์ พรอพเพอร์ตี้ส์ เมธอด ฟังก์ชัน รวมทั้งคีย์เวิร์ดต่างๆในวิชวลเบสิกสำหรับแอพพลิเคชันจะถูกบรรจุอยู่ใน on-line help ที่ว่านี้ทั้งหมด คุณสามารถเข้าถึงมันได้จากเมนู Help หรือจากการโฟกัสออปเจ็กต์ พรอพเพอร์ตี้หรือเมธอด แล้วกดปุ่ม F1 อันนี้จะไปเรียกหน้าต่างรายละเอียดของสิ่งที่คุณโฟกัสมาแสดงทันที



นอกจากฟีเจอร์ต่างๆที่สาธยายมาทั้งหมดแล้ว เอ็กเซลยังเสนอสิ่งที่เรียกกันว่า macro recorder ซึ่งจะทำหน้าที่เขียนโค้ดแมคโครให้กับคุณ คุณเพียงแต่เปิด recorder ขึ้นมา แล้วปฏิบัติสิ่งที่ต้องการให้โค้ดมันปฏิบัติ จากนั้นก็ปิด recorder เท่านี้แหละครับ เหลือเชื่อเลยเอ็กเซลมันจัดการสร้างโมดูลในวิชวลเบสิกสำหรับแอพพลิเคชันและใส่โค้ดเข้าไปโดยอัตโนมัติทันที ด้วยความสามารถดังกล่าวจะเห็นได้ว่าเจ้า recorder นี้ยังเป็นเครื่องมือในการเรียนรู้ที่ดีสำหรับผู้เริ่มต้นวิชวลเบสิกอีกด้วย

ถึงตรงนี้เห็นหรือยังครับว่าวิชวลเบสิกสำหรับแอพพลิเคชันนั้นให้ระบบการปรับปรุงแก้ไขและตรวจหาที่ผิดพลาดในโค้ดจุใจขนาดไหน มันเทียบเท่ากับเครื่องมือพัฒนาโปรแกรมทั่วไป แต่นับเป็นการเพิ่มความสามารถให้โปรแกรมสเปรดชีตอย่างที่ไม่มีหน้าไหนมาเทียบได้เลย

3. เอ็กเซลสนับสนุน OLE 2.0

เอ็กเซลสนับสนุนวิชวลอีดิต แดร็กแอนด์ดรอพข้ามแอพพลิเคชัน และฟีเจอร์ OLE กวาดตาหาให้ทั่วตลาดตอนนี้คุณจะไม่พบอะไรแบบนี้ในสเปรดชีตตัวอื่นเลย มันเป็นของดีที่สำคัญยิ่งยวดสำหรับผู้พัฒนาแอพพลิเคชันบนวินโดวส์ทั้งหลาย คงไม่ปฏิเสธนะครับว่า มันเป็นข่าวดีชิ้นโตต่อเรา เมื่อสามารถติดต่อกับแอพพลิเคชันอื่นในรูปแบบของ OLE ได้ง่ายๆ

ก่อนหน้า OLE นั้น เรามีเทคโนโลยีที่รู้จักกันในนามของ DDE (Dynamic Data Exchange) ซึ่งใครต่อใครพากันบ่นเรื่องความยุ่งยากในการใช้งาน เพราะมันต้องการคำสั่งเยอะแยะเพื่อที่จะสร้างช่องทางสื่อสารและผ่านคำสั่งจากแอพพลิเคชันหนึ่งไปยังแอพพลิเคชันหนึ่งอีกทีแถมท้ายด้วยข้อเสียเกี่ยวกับความซักช้าในการทำงานด้วยพีเจอร์นี้อีกหนึ่ง

OLE จึงกลายเป็นพระเอกหน้าใหม่ที่มาแทน DDE ทั้งด้านประสิทธิภาพความรวดเร็ว และทั้งด้านความสะดวกง่ายดายที่จะใช้งาน แทนที่คุณจะต้องไปนั่งป้อนคำสั่งข้ามช่องทางติดต่ออย่าง DDE คุณจะมีความรู้สึกเสมือนกำลังทำงานกับออปเจ็กต์ที่อยู่ในแอพพลิเคชันปัจจุบันเลยทีเดียว อย่างเช่น ถ้าต้องทำงานกับออปเจ็กต์ของไมโครซอฟต์โปรเจ็กต์ ซึ่งก็มีวิชวลเบสิกสำหรับแอพพลิเคชันเหมือนๆกับเอ็กเซล คุณก็สามารถเรียกใช้ออปเจ็กต์ที่ฝั่งโน้นด้วยภาษาแมคโครแบบใหม่นี่ผ่านพรอพเพอร์ตี้ส์และเมธอด แบบเดียวกับที่คุณเรียกใช้ออปเจ็กต์ในเอ็กเซลดีๆ นี่เอง คงจะมอง

ออกกันแล้วนะครับว่าเป็นการง่ายต่อการรวมสองให้เป็นหนึ่ง เราก็จะได้ฟีเจอร์ของหลายแอพพลิเคชันมาไว้ในที่เดียวกัน

4. เอ็กเซลสนับสนุนการสื่อสารข้อมูล WOSA

WOSA นั้นย่อมาจากคำว่า Windows Open Services Architecture และมักเน้นเกี่ยวกับเรื่องของมาตรฐานการถ่ายเทข้อมูล (data transfer protocols) ซึ่งแบ่งรูปแบบได้เป็น ODBC (Open Database Connectivity) MAPI (Microsoft's Messaging Applications Programming Interface) และ TAPI (Microsoft's Telephony Applications Programming Interface) มาดูกันในรายละเอียดเลยครับว่าแต่ละรูปแบบเป็นอย่างไร

ODBC:

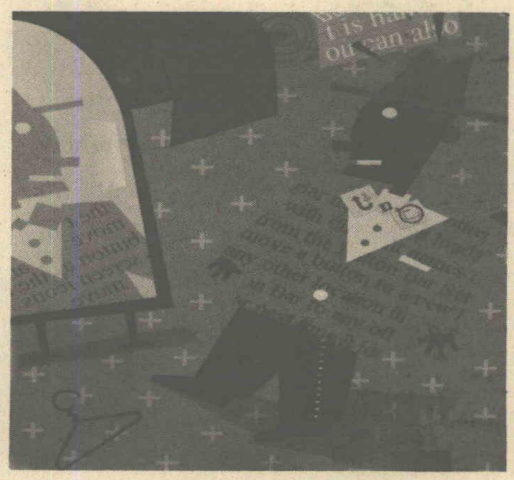
เป็นวิธีนำข้อมูลเข้าออกดาต้าเบสภายนอก ซึ่งสนับสนุน ODBC เอ็กเซลขายพร้อม ODBC มีไดรเวอร์ที่รู้จักดาต้าเบสฟอร์แมตได้หลากหลาย ไม่ว่าจะเป็น dBASE FoxPro MSAccess Oracle หรือ DEC RdB ถ้าแค่นี้ยังใช้ไม่ได้กับดาต้าเบสในงานของคุณ เราก็สามารถซื้อเพิ่มได้อีกนะครับ ซึ่งจะมีในรูปของ ADD-ON pack โดยซื้อได้จากตัวแทนจำหน่ายซึ่งเป็นแหล่งที่มาของเอ็กเซล 5.0 ในบ้านเรา

มีอยู่สองวิธีครับที่จะเอาข้อมูลเข้าออกดาต้าเบสภายนอกของคุณ คือใช้ Microsoft Query ซึ่งติดมากับเอ็กเซล 5.0 หรือใช้คำสั่งของ SQL ในวิชวลเบสิกสำหรับแอพพลิเคชัน การใช้ Microsoft Query นั้นเหมาะสำหรับผู้ใช้เป็นที่สุด มันทำให้รูปแบบการเรียกข้อมูลเป็นไปได้ง่ายดายสะดวกดีมาก เป็นในทำนองเดียวกันกับคอนเส็ปต์เกี่ยวกับฟรอนต์-เอนด์ใน MS Access ซึ่งมันเข้ากันได้ทั้งกับเอ็กเซลและไมโครซอฟต์เวิร์ดเสียด้วย

ส่วน SQL ในวิชวลเบสิกสำหรับแอพพลิเคชันนั้น จะให้โอกาสโปรแกรมเมอร์ในการใช้คำสั่ง SQL ที่ธรรมดาไปจนถึงขั้นซับซ้อนซ่อนเงื่อนผิดปกติ การที่คุณจะใช้คำสั่งในลักษณะนี้ได้ต้องมีไฟล์ XLODBC.XLA ซึ่งก็ให้มาพร้อมกับเอ็กเซล 5.0 เช่นกัน คำสั่ง SQL ในวิชวลเบสิกสำหรับแอพพลิเคชันจะถูกใช้ในการรับส่งข้อมูลได้โดยตรงจากเวอร์กชีตหรือตัวแปรอะเรย์ของแมคโคร

MAPI:

อันนี้ทำให้เป็นไปได้ที่แอพพลิเคชันสร้างเองจากเอ็กเซลของคุณจะไปรวมตัวกับฟังก์ชัน electronic messaging ในระบบอีเมล์ซึ่งสนับสนุน MAPI ซึ่งอย่างนี้ก็หมายความว่าแอพพลิเคชันของคุณสามารถรับส่งข้อมูลในเวอร์กชีตผ่านระบบสื่อสารดังกล่าว จะกำหนดผู้รับ จะส่งสารอันใด หรือจะใช้ฟีเจอร์ทั้งหลายในระบบอย่างไรก็ได้ การใช้ฟังก์ชัน MAPI นั้นต้องอาศัยไฟล์ๆหนึ่งชื่อ XLMAPI.XLL มาประกอบ ซึ่งนี่ก็ไม่ต้องไปเสาะแสวงจากไหนไกล เขาพกติดมาให้พร้อมเอ็กเซลครับ



TAPI:

คุณสามารถเอาข้อมูลชนิด real-time ผ่านโมเด็มเข้าแอพพลิเคชันของเอ็กเซลได้ก็ด้วยฟีเจอร์นี้ เป็นการเหมาะมากสำหรับการถ่ายทอดข้อมูลทางไกล

5. เอ็กเซลเป็นผู้นำตลาดสเปรดชีต

คราวนี้มาพูดถึงเหตุผลข้อสุดท้ายที่ทำให้คุณน่าจะอยากซื้อเอ็กเซล 5.0 มาใช้ในบริษัท นั่นก็คือ เครดิตและความน่าเชื่อถือของสเปรดชีตของไมโครซอฟต์ เห็นได้จากที่ขายดิบขายดีเป็นอันดับหนึ่งของโลก ไม่ว่าจะบนแพลตฟอร์มไหน วินโดวส์ แมคอินทอช หรือดอสก็แล้วแต่ จากตัวเลขที่สำรวจเมื่อไม่นานมานี้ เชื่อได้ว่ามีเครื่องผู้ใช้ (อย่างถูก,ต้องตามกฎหมาย) ติดตั้งเอ็กเซลไม่น้อยกว่าเจ็ดล้านเครื่อง

ข้อดีนี้แม้จะเป็นตัวแปรสุดท้าย แต่ก็สลักสำคัญไม่ใช่น้อยเลยนะครับ ถึงแม้แอพพลิเคชันจะเลิศเลอ ทรงประสิทธิภาพสักปานใด แต่ถ้าหากคุณทำออกมาแล้วไม่มีเครื่องผู้ใช้ส่วนใหญ่รับได้ มันก็ไม่มีประโยชน์อันใดขึ้นมา

สำหรับคุณที่ต้องการข้อมูลเกี่ยวกับสเปรดชีตแบบที่จะนำมาช่วยพิจารณาซื้อผลิตภัณฑ์เข้าบริษัท ก็เห็นจะเพียงพอแค่นี้ ตอนต่อไปผมจะนำรายละเอียดอันเป็นประโยชน์ต่อโปรแกรมเมอร์มาเสนอ ซึ่งถ้าคุณสนใจแล้วยังไม่ค่อยมีพื้นเกี่ยวกับวิชวลเบสิก ขอแนะนำให้ลองหันมาศึกษาไว้แต่เนิ่นๆ นะครับ ก็จากวารสารไมโครคอมพิวเตอร์นี่แหละ *สวัสดีครับ*